# Einführung in Visual Computing

186.822

# Freeform Surfaces

Werner Purgathofer

# Lehrveranstaltungsbewertung

- Studierende werden ersucht, besuchte Lehrveranstaltungen zu bewerten (in TISS).
- ich wurde gebeten meine Studierenden daran zu erinnern, sich aktiv an der Lehrveranstaltungsbewertung zu beteiligen.
- Danke!

# Freeform Surfaces in the Rendering Pipeline

*object capture/creation*

scene objects in object space

*modeling*

*viewing*

*projection*

vertex stage („vertex shader“)

transformed vertices in clip space

*clipping + homogenization*

scene in normalized device coordinates

*viewport transformation*

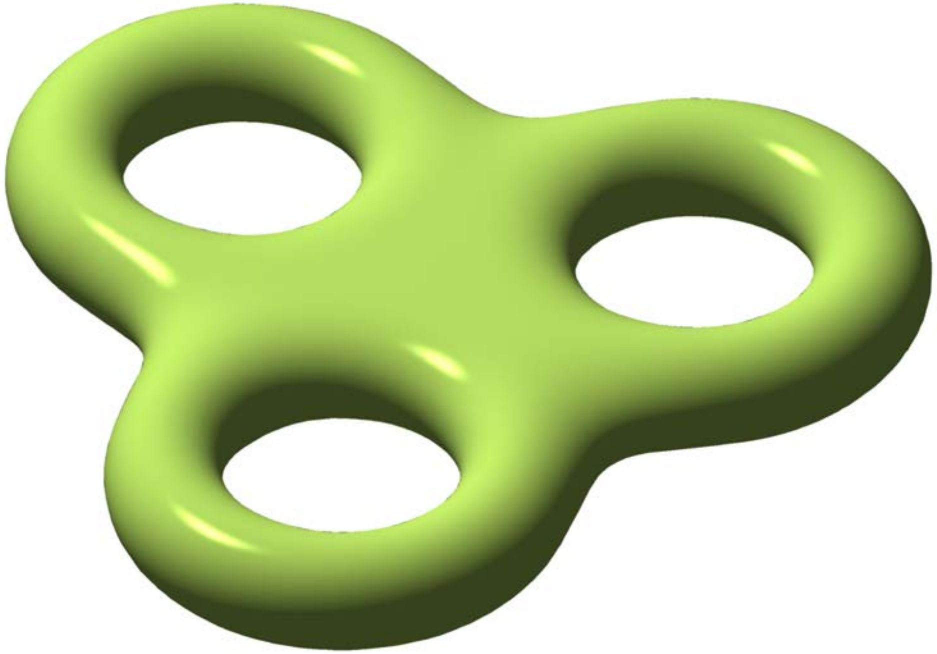
*rasterization*

*shading*

pixel stage („fragment shader“)

raster image in pixel coordinates

# Curved Lines and Surfaces



defined by

- mathematical functions (implicit, explicit, parametrically)
- set of data points (surface fitting)

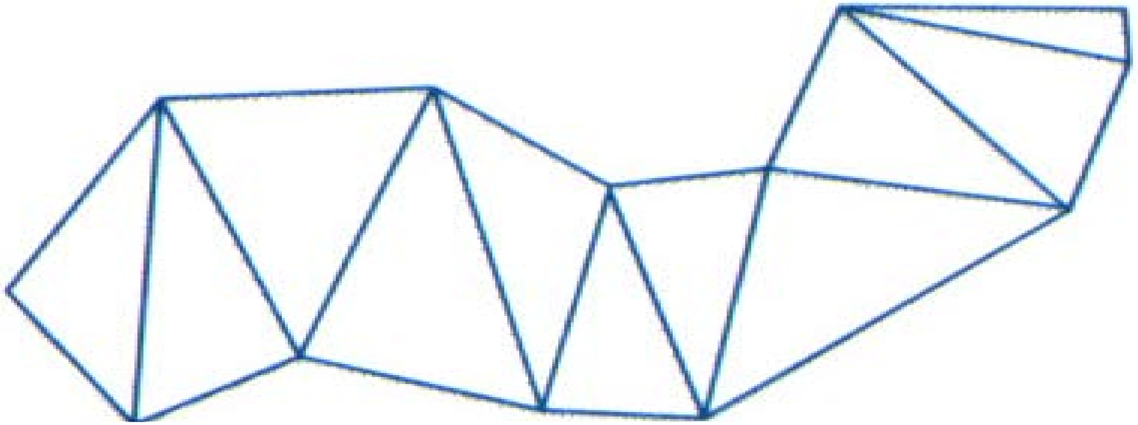
***tesselation*** to get polygon mesh approximation

- triangles
- quadrilaterals ... (planar?!)

efficient data structures for tiled surfaces

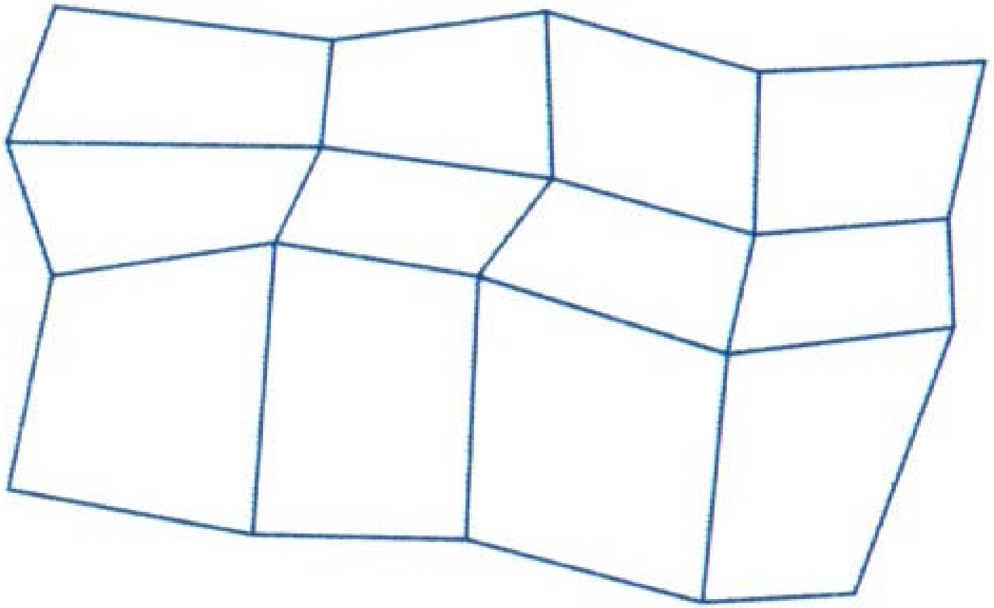
**triangle strip**

- n – 2 triangles for n vertices



**quadrilateral mesh**

- (n – 1) x (m – 1) quadrilaterals



# Nonparametric ↔ Parametric

$$y = f(x)$$

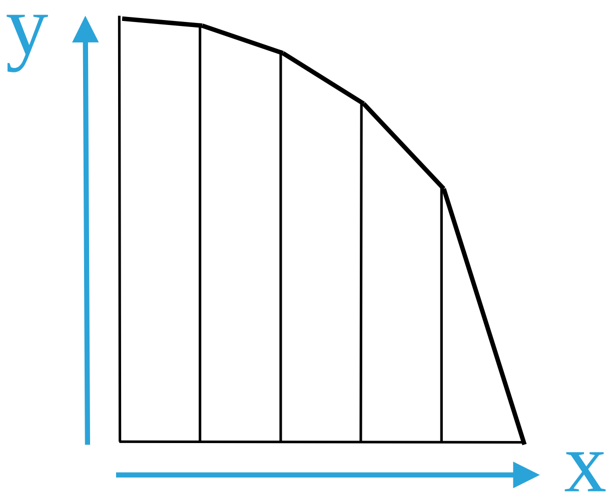
$$x = f(u)$$
$$y = g(u)$$

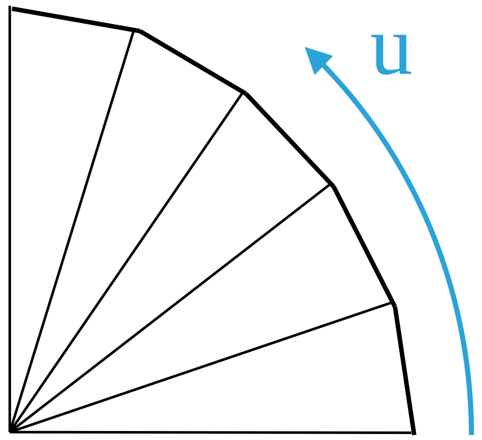
*axis dependent*

*axis independent*

*example:* $y = \sqrt{1-x^2}$

$x=\cos(u) \quad y=\sin(u)$

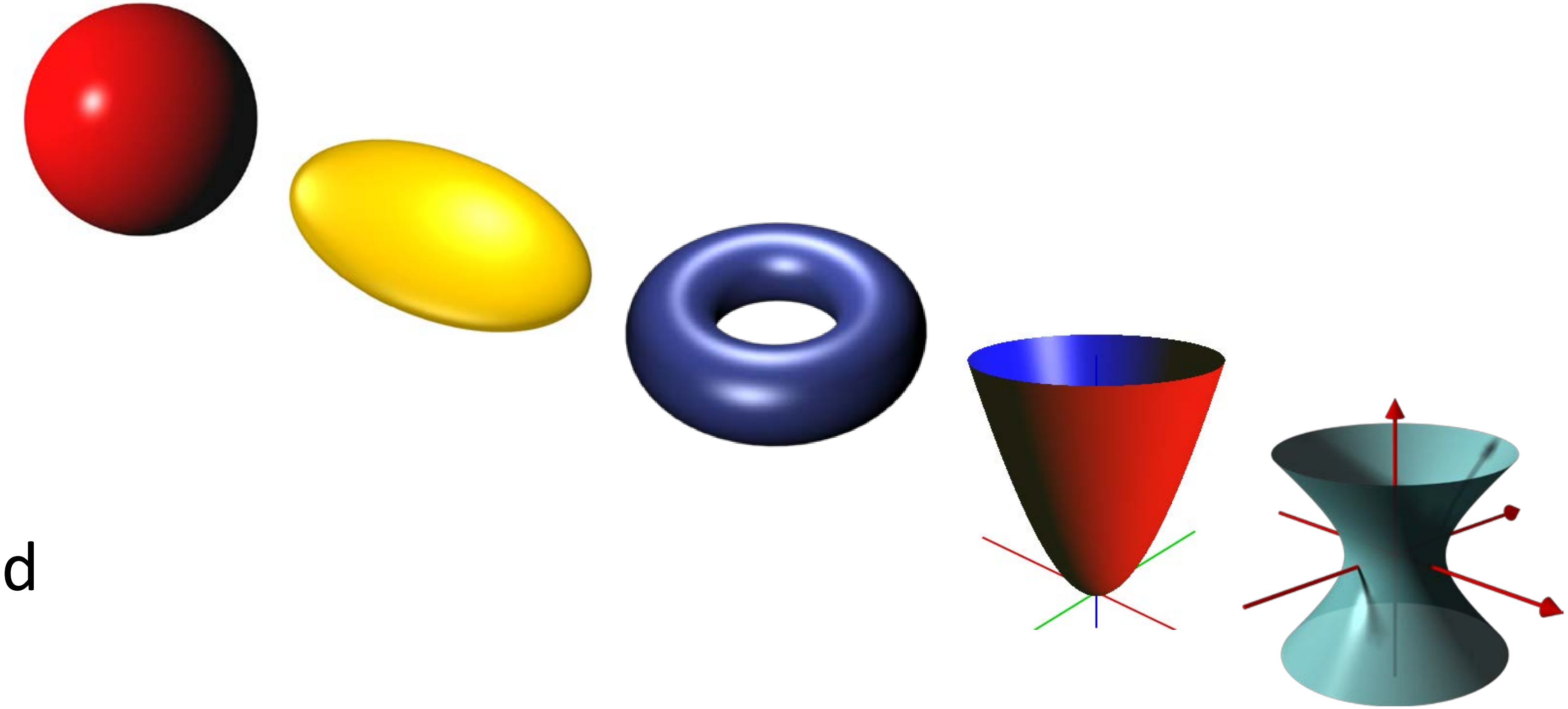


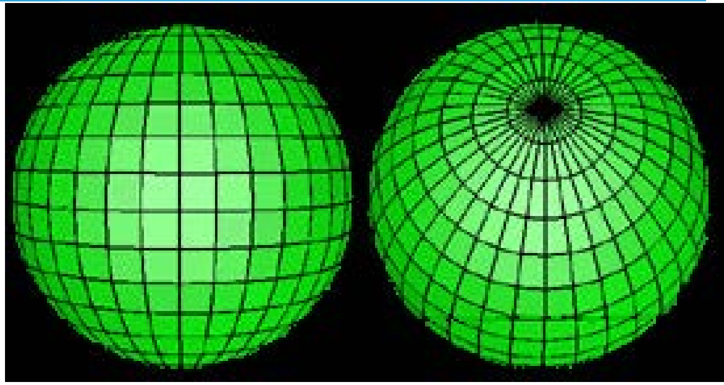
# Quadric Surfaces

defined by second degree equations (quadrics)

- sphere
- ellipsoid
- torus
- paraboloid
- hyperboloid
- ...



# Quadric Surfaces: Sphere

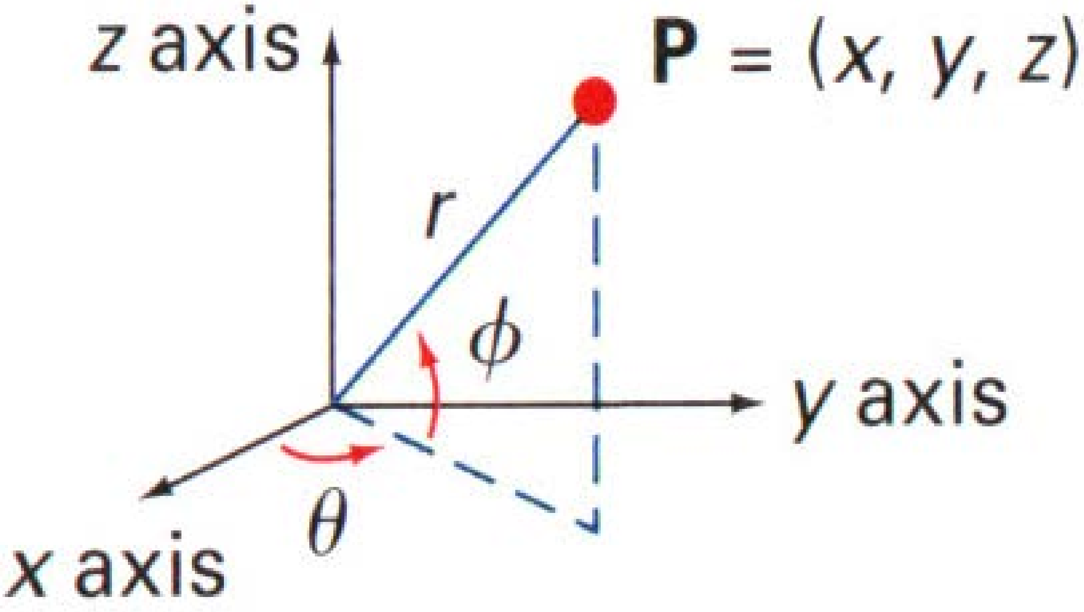


**implicit:** $\boxed{x^2 + y^2 + z^2 = r^2}$

**parametric:**

$$x = r \cos\phi \cos\theta, \quad -\pi/2 \le \phi \le \pi/2$$
$$y = r \cos\phi \sin\theta, \quad -\pi \le \theta \le \pi$$
$$z = r \sin\phi$$

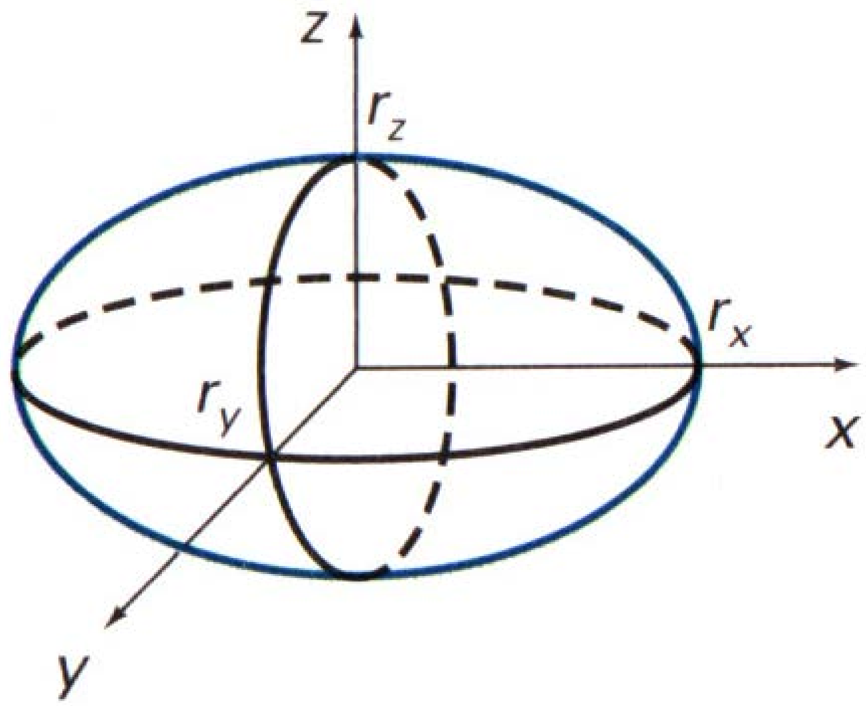
*parametric coordinate position $(r, \theta, \phi)$ on the surface of a sphere with radius r*



# Quadric Surfaces: Ellipsoid

**implicit:**

$$\left(\frac{x}{r_x}\right)^2 + \left(\frac{y}{r_y}\right)^2 + \left(\frac{z}{r_z}\right)^2 = 1$$



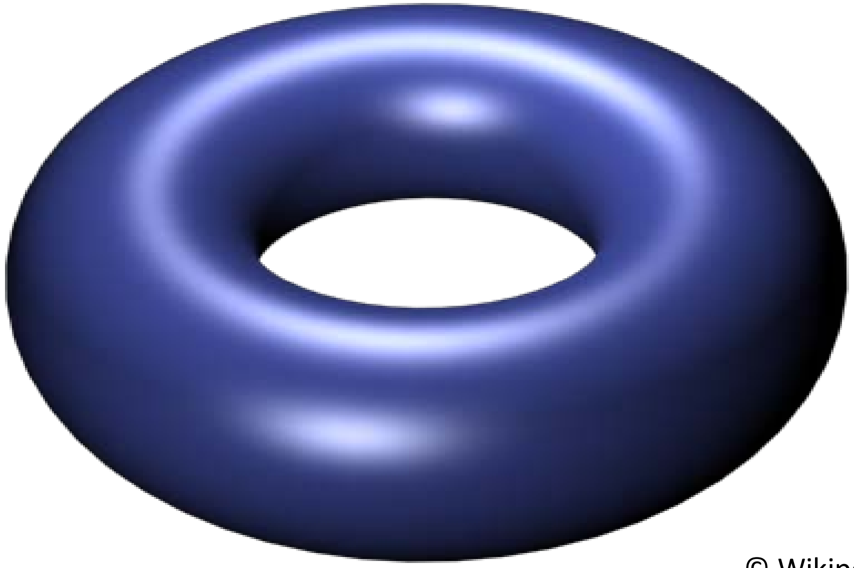

**parametric:**

$$\begin{aligned} x &= r_x \cos\phi \cos\theta, & -\pi/2 \le \phi \le \pi/2 \\ y &= r_y \cos\phi \sin\theta, & -\pi \le \theta \le \pi \\ z &= r_z \sin\phi \end{aligned}$$

# Quadric Surfaces: Torus

**implicit:**

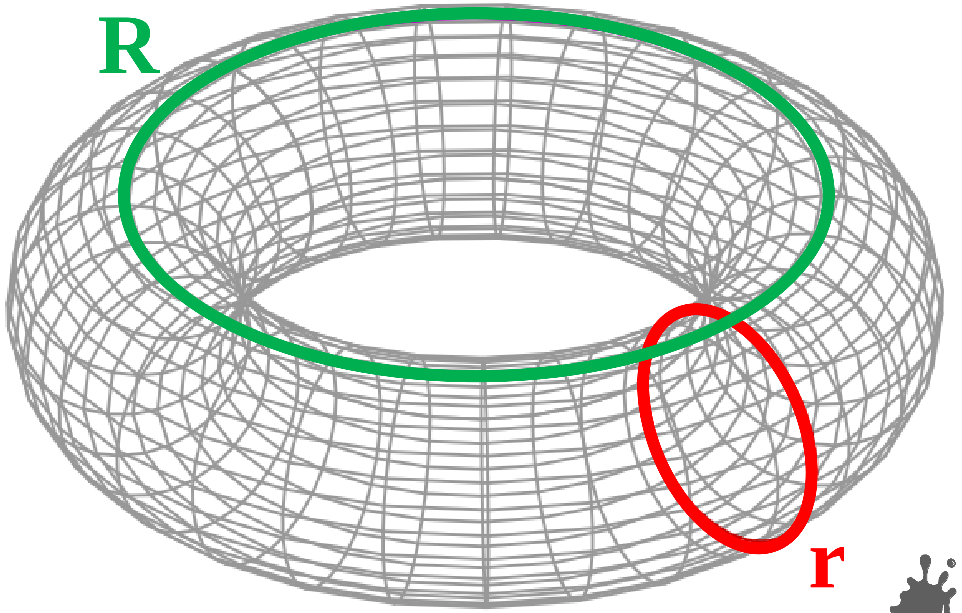
$$(R - \sqrt{x^2 + y^2}\,)^2 + z^2 = r^2$$



© Wikipedia

**parametric:**

$$x = (R + r\cos\phi)\cos\theta, \qquad -\pi \le \phi \le \pi$$
$$y = (R + r\cos\phi)\sin\theta, \qquad -\pi \le \theta \le \pi$$
$$z = r\sin\phi$$



# Properties of Curves

- **possible curve forms**
- **interpolating or approximating** control points?
- **global or local influence** of control points?
- **multiple points** possible? (for closed curves and corners)
- **degree of continuity** at concatenations (C,G)?
- **oscillatory behavior** - compact or overshooting?
- **axis (in)dependence?** (does the curve change when the coordinate system is rotated?)

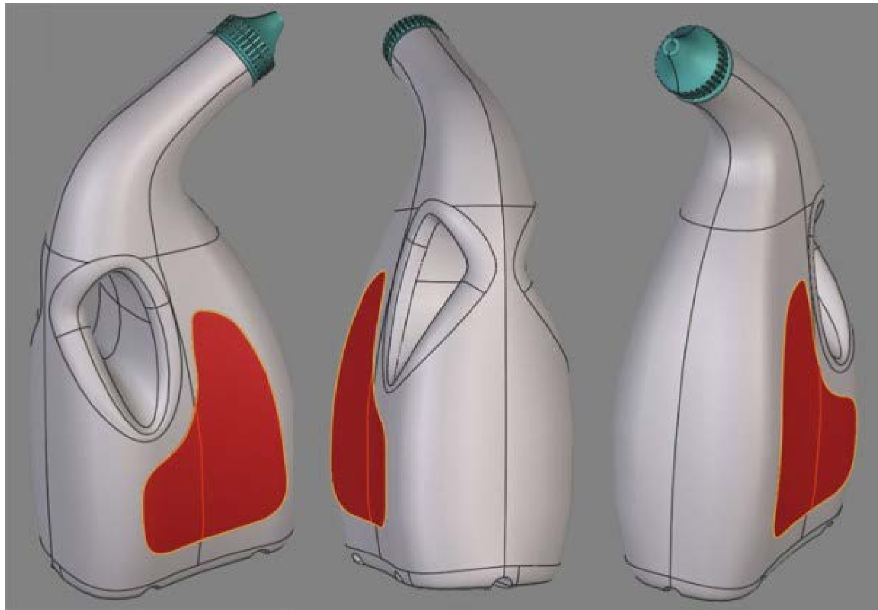
# Spline Representations

**spline curve**

- composite curve
- polynomial sections, piecewise continuous
- continuity conditions

**spline surface**

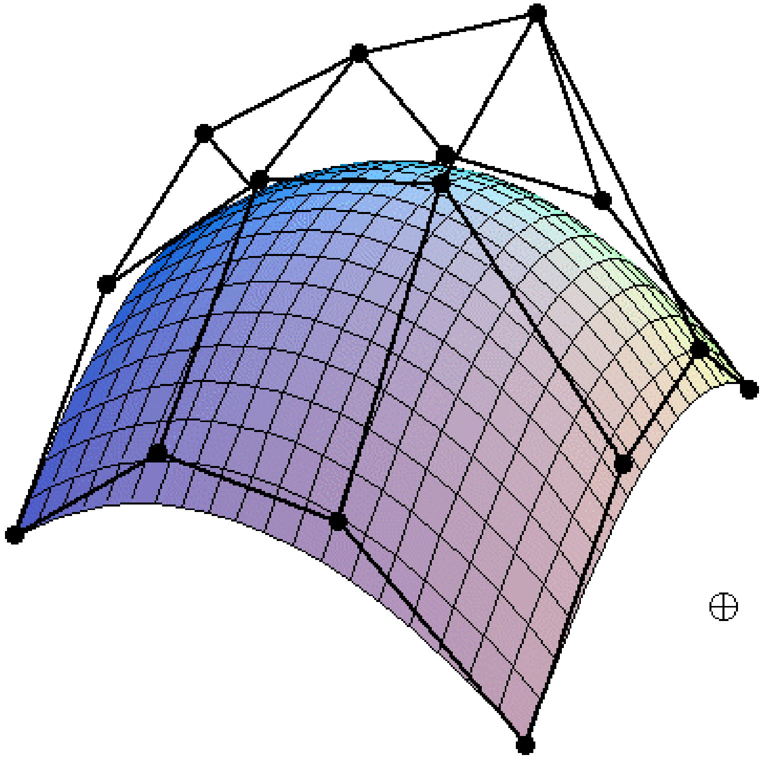
- two sets of orthogonal spline curves

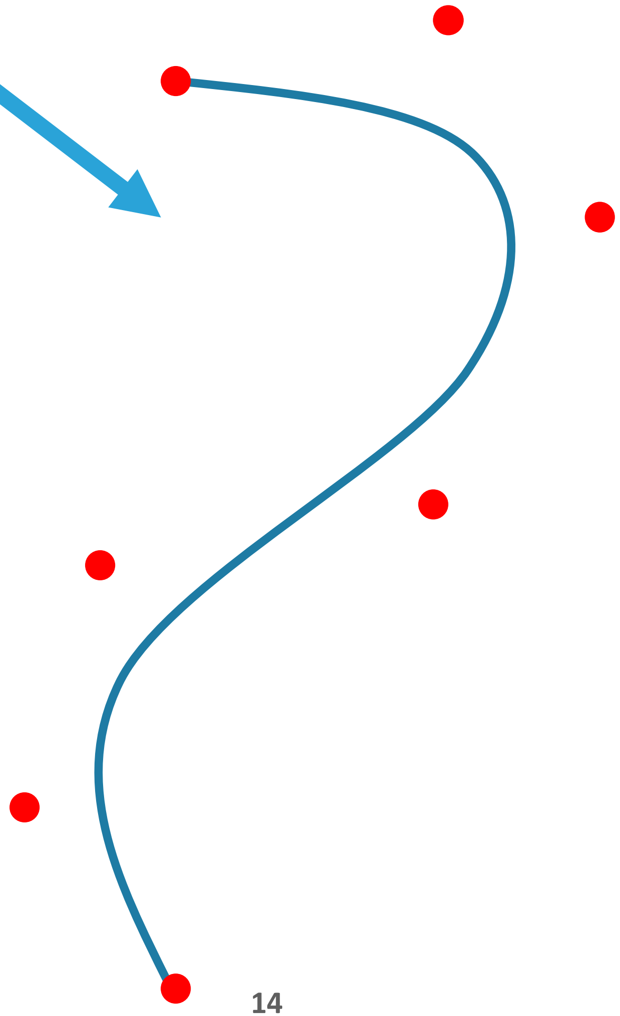



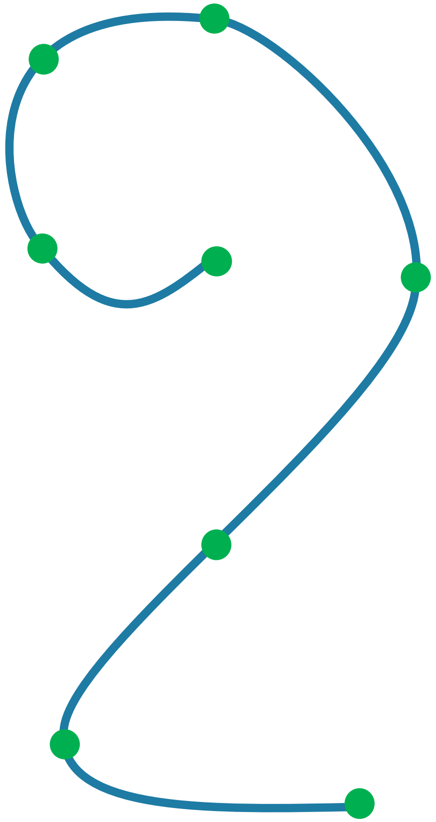
# Spline Curves

spline specification with control points

- interpolating splines
- approximating splines



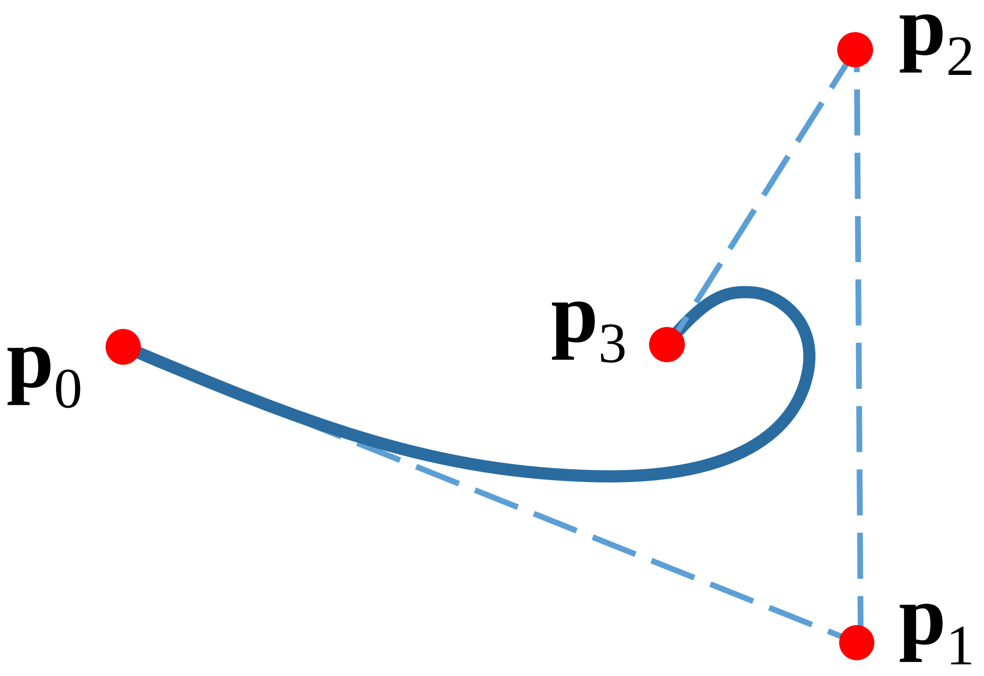


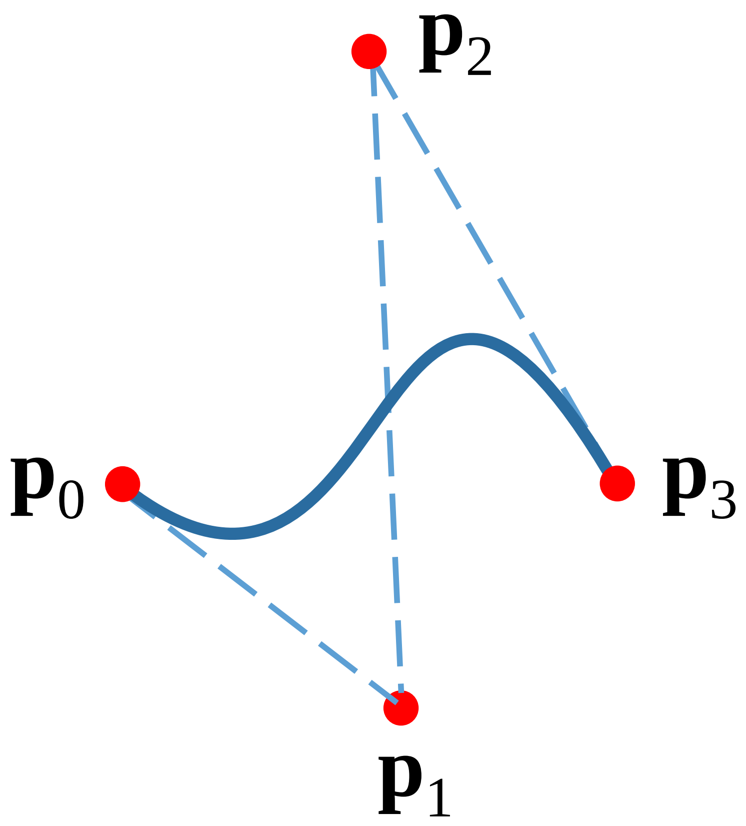


# Splines: Control Polygon

(also called „**Characteristic Polygon**“)

polygon defining the curve

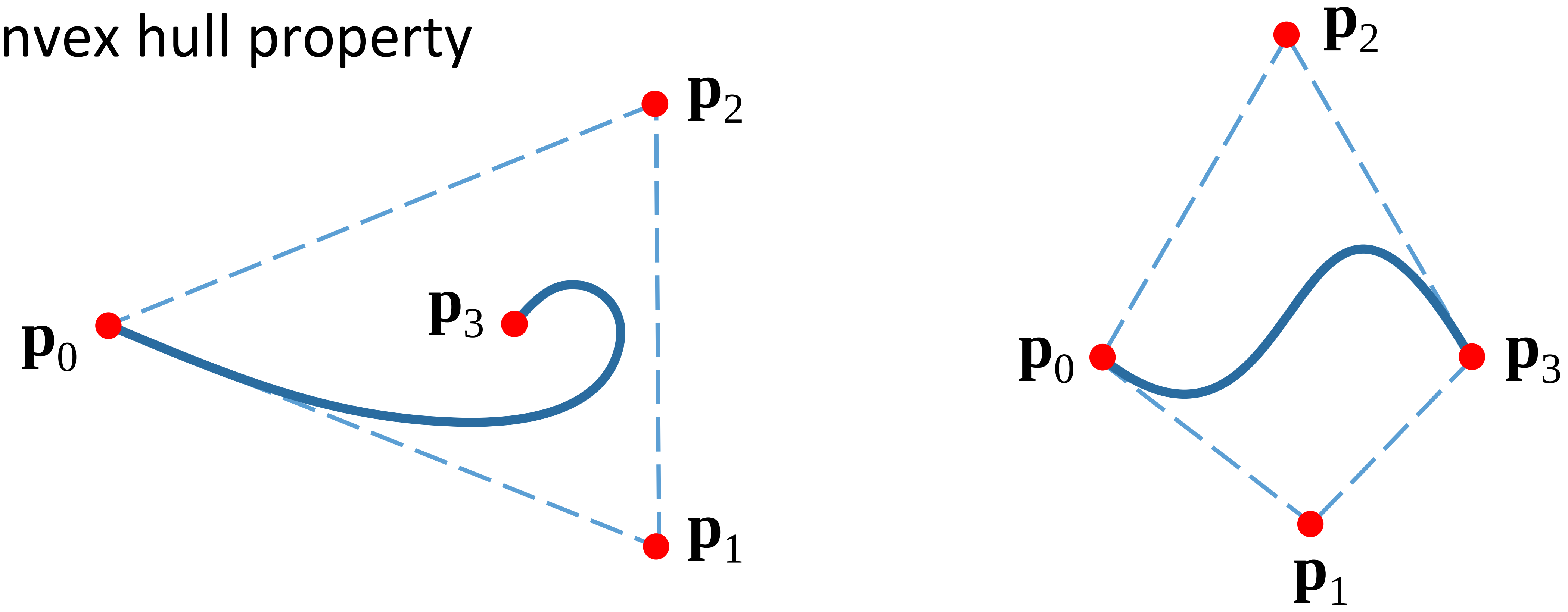


operations on splines

- move, insert control points
- spline transformation by transforming all control points

convex hull property

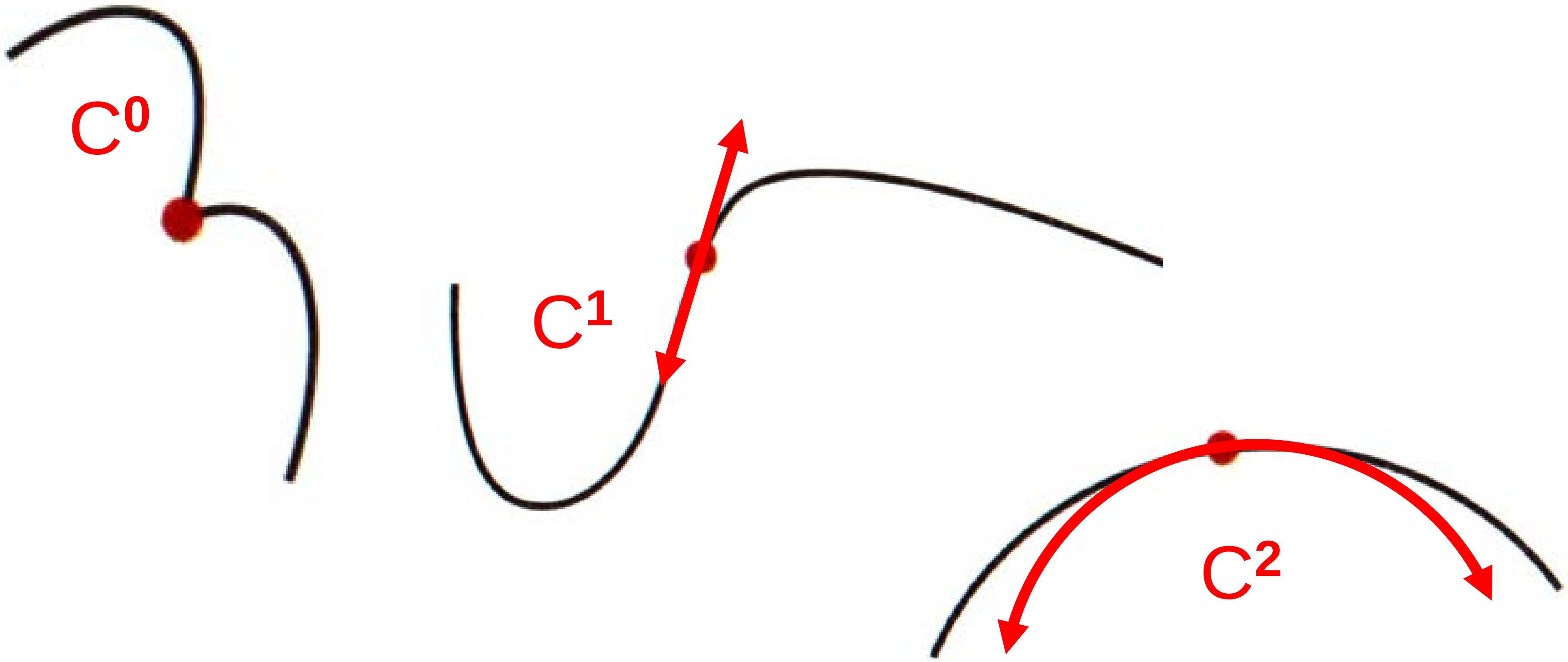


## ***parametric* continuity conditions** ($C^n$)

- derivations at section joints are equal

$$x = x(u) \qquad y = y(u) \qquad z = z(u) \qquad u_1 \le u \le u_2$$

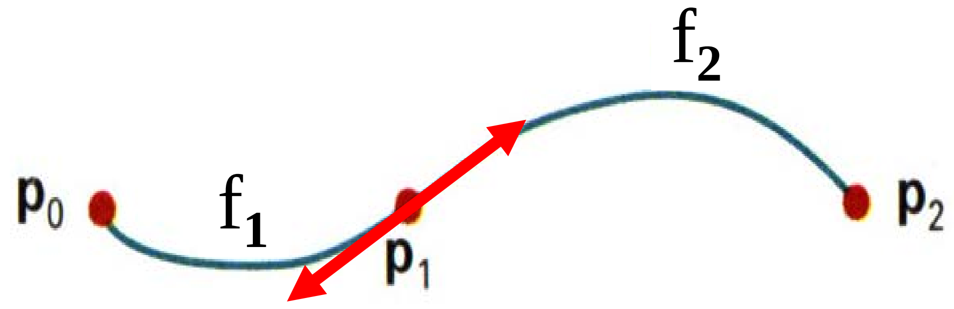
- $C^0$ continuity
- $C^1$ continuity
- $C^2$ continuity

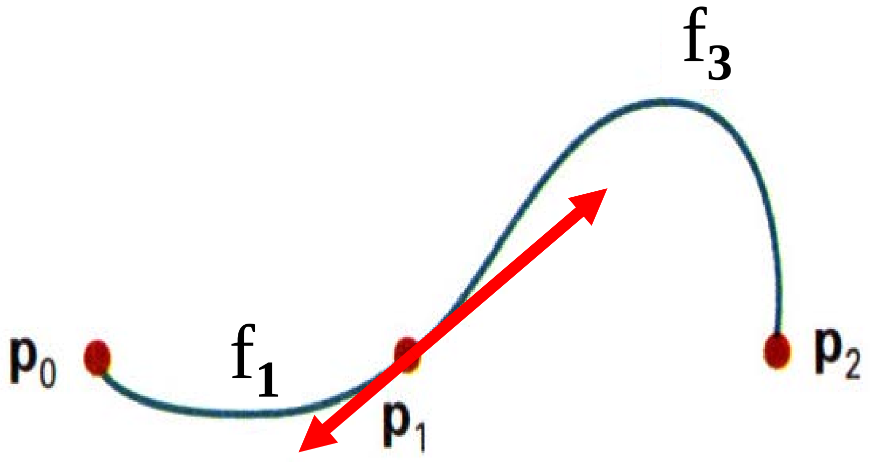


# Spline: Continuity Conditions (2)

***geometric* continuity conditions** ($G^n$)

- derivations at joints have different magnitudes
- $G^0$ (=$C^0$) continuity
- $G^1$ continuity (tangent vectors are collinear)
- $G^2$ continuity …
- weaker than $C^n$




*tangent vector of* $f_3$ *at* $p_1$ *has a greater magnitude than the tangent vector of* $f_1$ *at* $p_1$

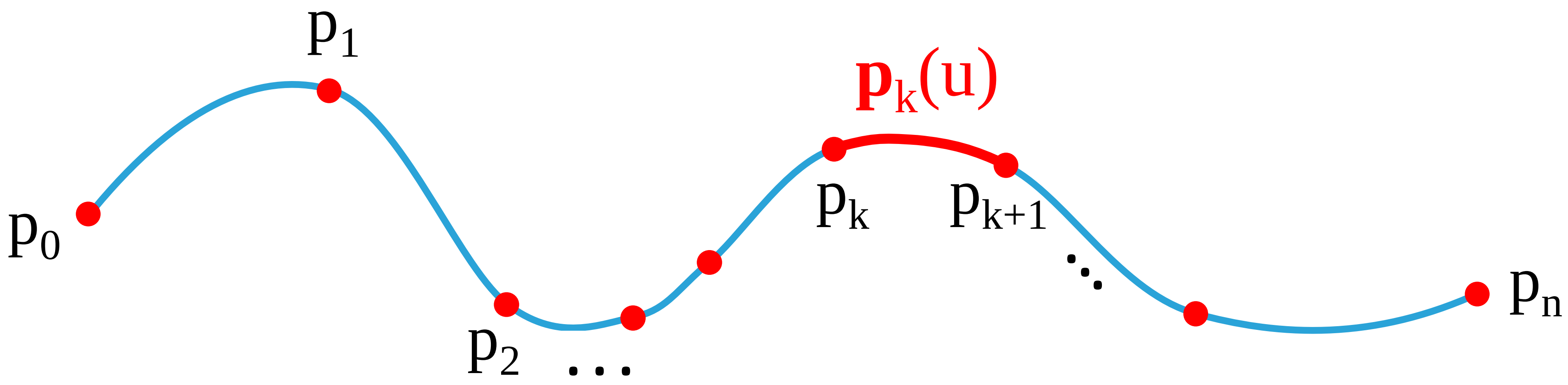
# Cubic Spline Interpolation

… has n+1 control points $\mathbf{p_i} = (x_i, y_i, z_i) \quad i = 0, 1, 2, \ldots, n$

cubic polynomial $\mathbf{p}_k(u)$ between pair $(\mathbf{p}_k , \mathbf{p}_{k+1})$ of control points

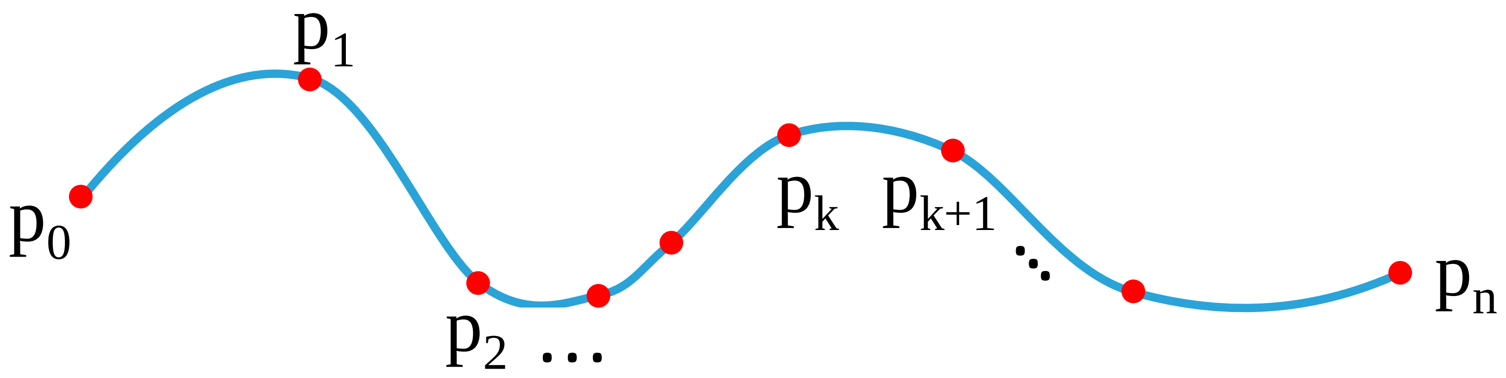
$$\mathbf{p_k}(u) = \mathbf{a_k}u^3 + \mathbf{b_k}u^2 + \mathbf{c_k}u + \mathbf{d_k}$$

$$k = 0, 1, 2, \ldots, n-1, \qquad 0 \leq u \leq 1$$



# Natural Cubic Splines

- adjacent curve segments: same 1st & 2nd derivative ($C^2$ continuity)
- solving an equation system with 4n variables
- two extra conditions required (e.g., $\mathbf{p_0}''(0) = 0, \quad \mathbf{p_{n\text{-}1}}''(1) = 0$ )
- global influence of control points




$$\mathbf{p}_i(0) = \mathbf{p}_i, \qquad i = 0,\ldots,n-1$$

$$\mathbf{p}_i(1) = \mathbf{p}_{i+1}, \qquad i = 0,\ldots,n-1$$

$$\mathbf{p}_i'(1) = \mathbf{p}_{i+1}'(0), \qquad i = 0,\ldots,n-2$$

$$\mathbf{p}_i''(1) = \mathbf{p}_{i+1}''(0), \qquad i = 0,\ldots,n-2$$

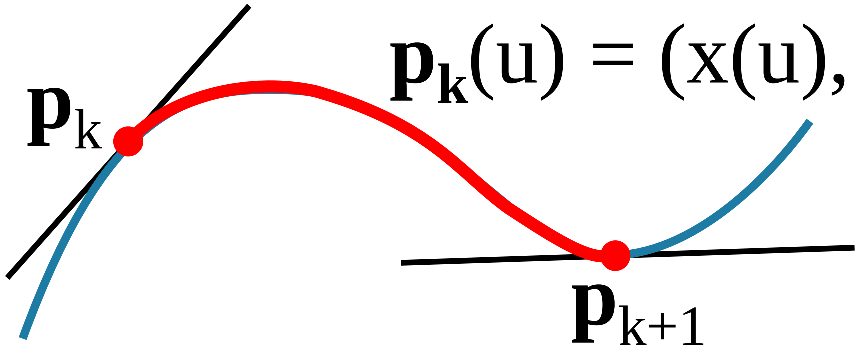
# Hermite Interpolation (1)

tangent $\mathbf{Dp}_{\mathbf{k+1}}$ specified at each control point

→ local influence of control points

$$\begin{aligned} \mathbf{p}_{\mathbf{k}}(0) &= \mathbf{p}_{k} \\ \mathbf{p}_{\mathbf{k}}(1) &= \mathbf{p}_{k+1} \\ \mathbf{p}'_{\mathbf{k}}(0) &= \mathbf{Dp}_{k} \\ \mathbf{p}'_{\mathbf{k}}(1) &= \mathbf{Dp}_{k+1} \end{aligned}$$

$k = 0, \ldots, n-1$



# Hermite Interpolation (2)

$$\mathbf{p_k}(u) = \mathbf{a_k}u^3 + \mathbf{b_k}u^2 + \mathbf{c_k}u + \mathbf{d_k} \qquad 0 \le u \le 1$$

$$\mathbf{p_k}(0) = \mathbf{p}_k$$
$$\mathbf{p_k}(1) = \mathbf{p}_{k+1}$$
$$\mathbf{p'_k}(0) = \mathbf{Dp}_k$$
$$\mathbf{p'_k}(1) = \mathbf{Dp}_{k+1}$$

$$\mathbf{p}_k(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{a_k} \\ \mathbf{b_k} \\ \mathbf{c_k} \\ \mathbf{d_k} \end{bmatrix} \qquad \mathbf{p}_k'(u) = \begin{bmatrix} 3u^2 & 2u & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{a_k} \\ \mathbf{b_k} \\ \mathbf{c_k} \\ \mathbf{d_k} \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{p_k} \\ \mathbf{p_{k+1}} \\ \mathbf{Dp_k} \\ \mathbf{Dp_{k+1}} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{a_k} \\ \mathbf{b_k} \\ \mathbf{c_k} \\ \mathbf{d_k} \end{bmatrix}$$

# Hermite Interpolation (3)

$$\begin{bmatrix} \mathbf{a}_k \\ \mathbf{b}_k \\ \mathbf{c}_k \\ \mathbf{d}_k \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}^{-1} \cdot \begin{bmatrix} \mathbf{p}_k \\ \mathbf{p}_{k+1} \\ \mathbf{Dp}_k \\ \mathbf{Dp}_{k+1} \end{bmatrix} = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{p}_k \\ \mathbf{p}_{k+1} \\ \mathbf{Dp}_k \\ \mathbf{Dp}_{k+1} \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{a}_k \\ \mathbf{b}_k \\ \mathbf{c}_k \\ \mathbf{d}_k \end{bmatrix} = M_H \cdot \begin{bmatrix} \mathbf{p}_k \\ \mathbf{p}_{k+1} \\ \mathbf{Dp}_k \\ \mathbf{Dp}_{k+1} \end{bmatrix}$$

“Hermite matrix”

# Hermite Interpolation (4)

$$\mathbf{p}_k(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{a}_k \\ \mathbf{b}_k \\ \mathbf{c}_k \\ \mathbf{d}_k \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{a}_k \\ \mathbf{b}_k \\ \mathbf{c}_k \\ \mathbf{d}_k \end{bmatrix} = M_H \cdot \begin{bmatrix} \mathbf{p}_k \\ \mathbf{p}_{k+1} \\ \mathbf{Dp}_k \\ \mathbf{Dp}_{k+1} \end{bmatrix}$$

$$\mathbf{p}_k(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \cdot M_H \cdot \begin{bmatrix} \mathbf{p}_k \\ \mathbf{p}_{k+1} \\ \mathbf{Dp}_k \\ \mathbf{Dp}_{k+1} \end{bmatrix}$$

# Hermite Interpolation (5)

$$\mathbf{p}_k(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{p}_k \\ \mathbf{p}_{k+1} \\ \mathbf{Dp}_k \\ \mathbf{Dp}_{k+1} \end{bmatrix}$$

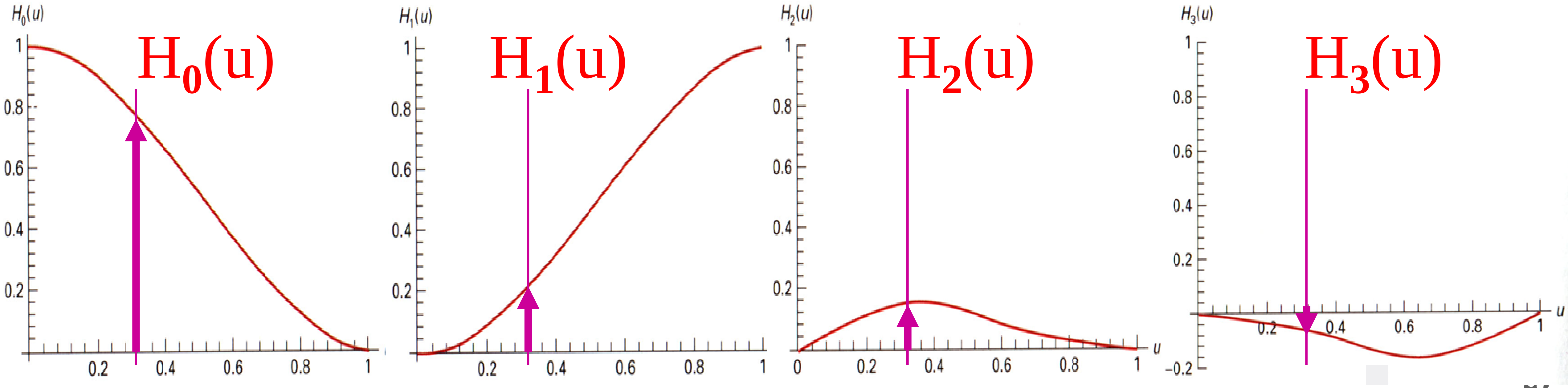
$$\mathbf{p}_k(u) = \mathbf{p}_k(2u^3 - 3u^2 + 1) + \mathbf{p}_{k+1}(-2u^3 + 3u^2) + \mathbf{Dp}_k(u^3 - 2u^2 + u) + \mathbf{Dp}_{k+1}(u^3 - u^2)$$

# Hermite Interpolation (6)

$$\mathbf{p}_k(u) = \mathbf{p}_k(2u^3 - 3u^2 + 1) + \mathbf{p}_{k+1}(-2u^3 + 3u^2) + \mathbf{Dp}_k(u^3 - 2u^2 + u) + \mathbf{Dp}_{k+1}(u^3 - u^2)$$

$$\mathbf{p}_k(u) = \mathbf{p}_k H_0(u) + \mathbf{p}_{k+1} H_1(u) + \mathbf{Dp}_k H_2(u) + \mathbf{Dp}_{k+1} H_3(u)$$

$H_k(u)$ blending functions:



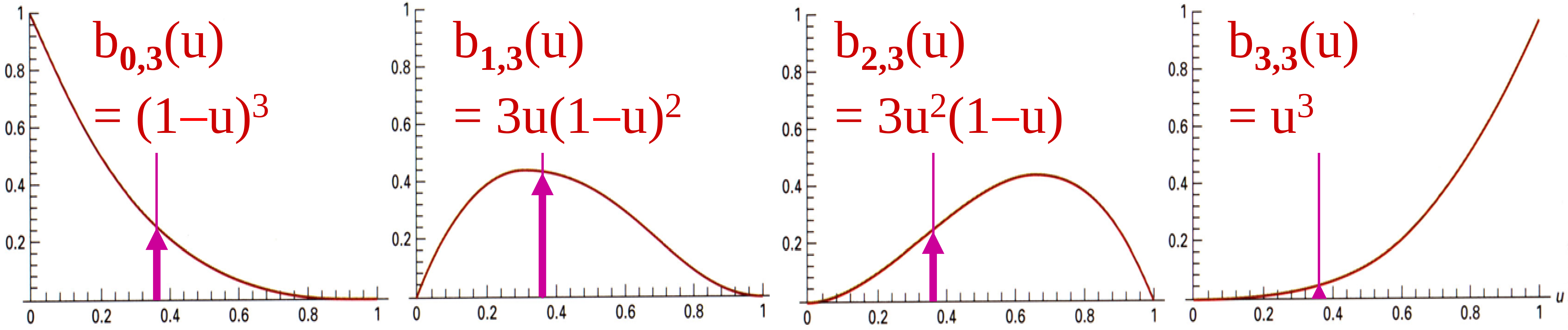
= spline approximation for points $\mathbf{p_i}$ , $i = 0, \ldots, n$

$$\mathbf{p}(u) = \sum_{k=0}^{n} \mathbf{p_k}\, b_{k,n}(u) \qquad 0 \le u \le 1$$

Bernstein polynomials

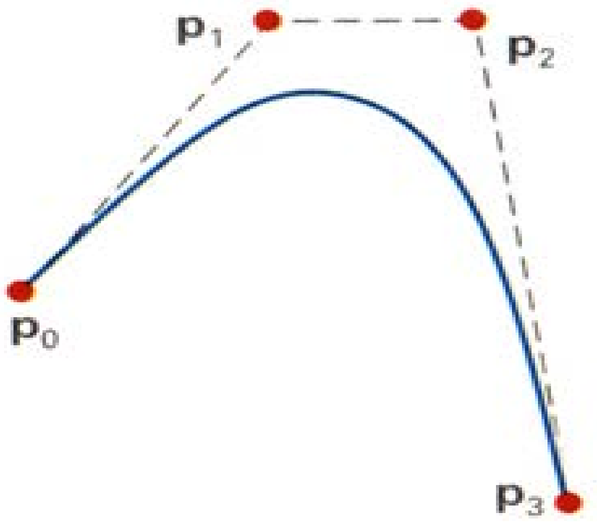
$$b_{k,n}(u) = \binom{n}{k} u^k (1-u)^{n-k}$$

# Cubic Bézier Blending Functions



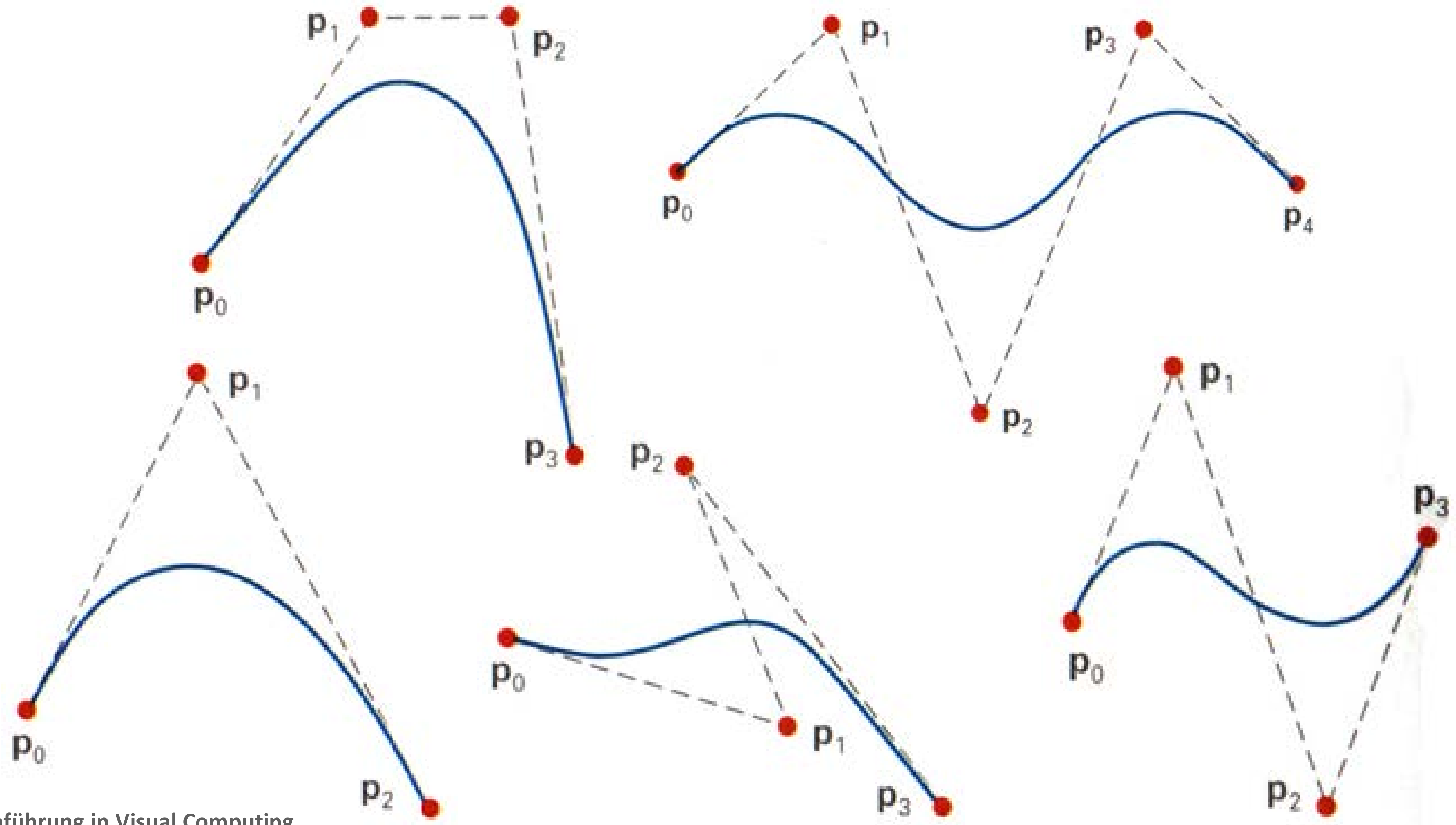

$$\mathbf{p}(u) = (1-u)^3 \cdot \mathbf{p_0} + 3u(1-u)^2 \cdot \mathbf{p_1} + 3u^2(1-u) \cdot \mathbf{p_2} + u^3 \cdot \mathbf{p_3}$$

*the 4 Bézier blending functions for cubic curves (n=3)*



# 2-Dimensional Bézier Curves Examples

generated from 3, 4, and 5 control points



# Bézier Curves Properties

- $\mathbf{p}(u)$ polynomial of degree n, global influence
- $\mathbf{p}(u)$ interpolates start and endpoint

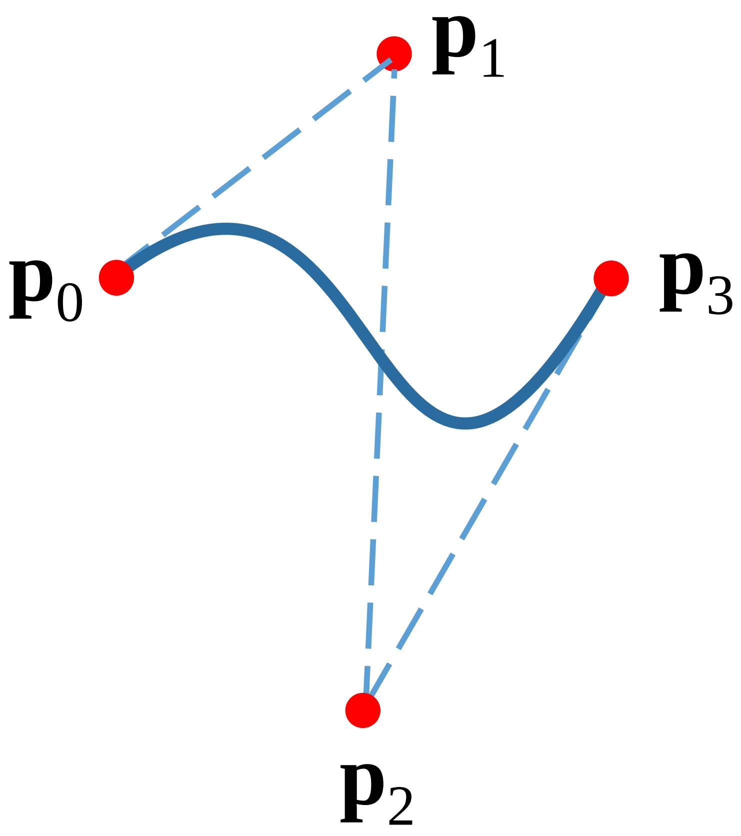
$$\mathbf{p}(0) = p_0\,, \quad \mathbf{p}(1) = p_n$$

- tangents at start and endpoint

$$\mathbf{p}'(0) = -n\mathbf{p_0} + n\mathbf{p_1}$$
$$\mathbf{p}'(1) = -n\mathbf{p_{n-1}} + n\mathbf{p_n}$$

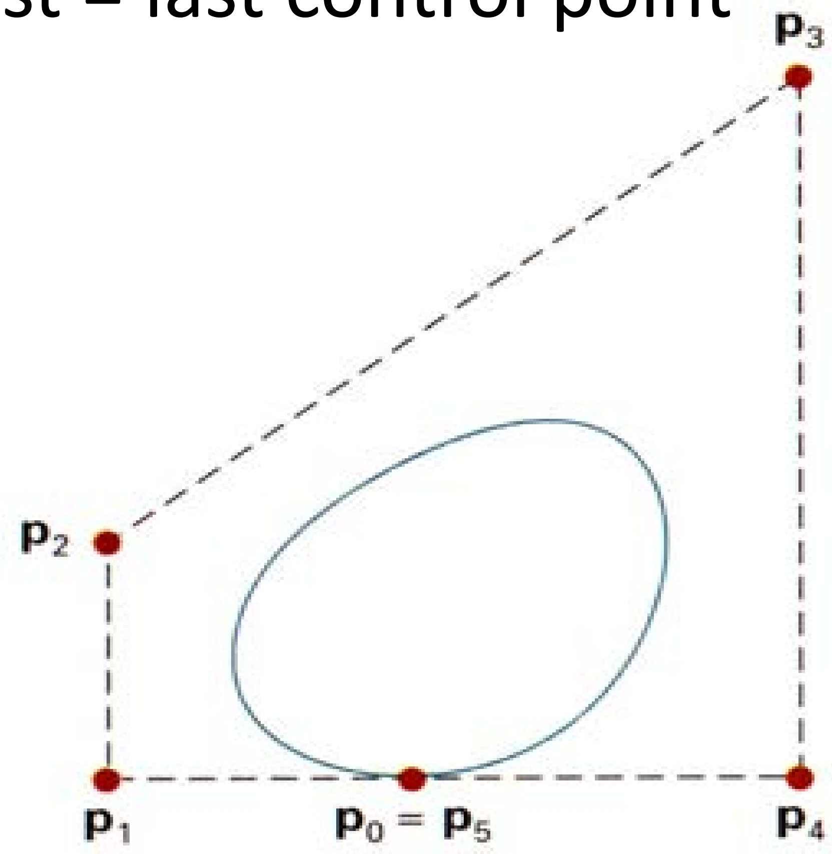
- convex hull property

$$\sum_{k=0}^{n} b_{k,n}(u) = 1$$

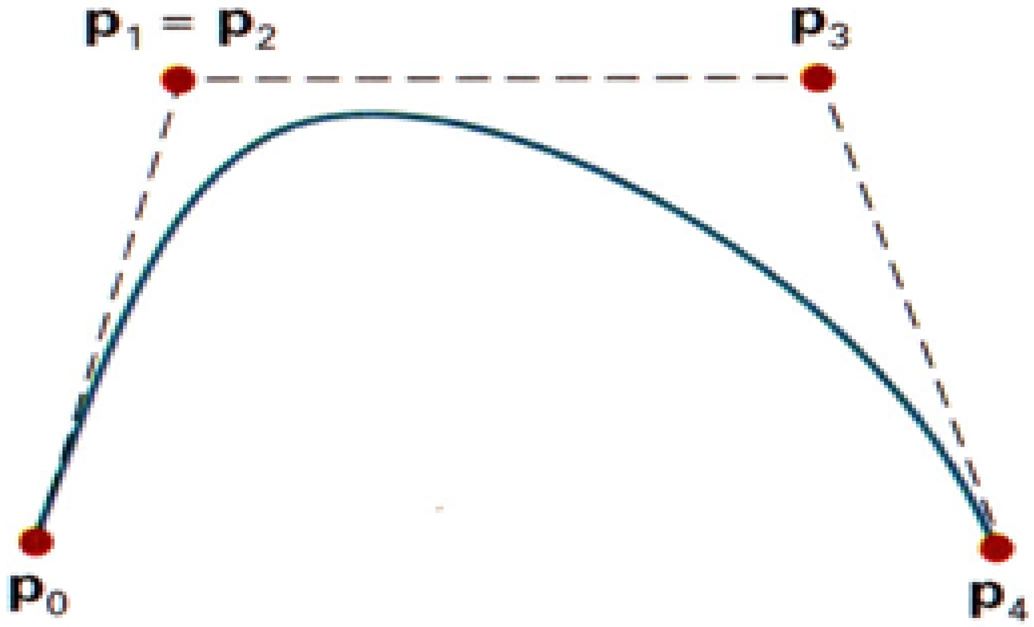


# Bézier Curves Design Techniques (1)

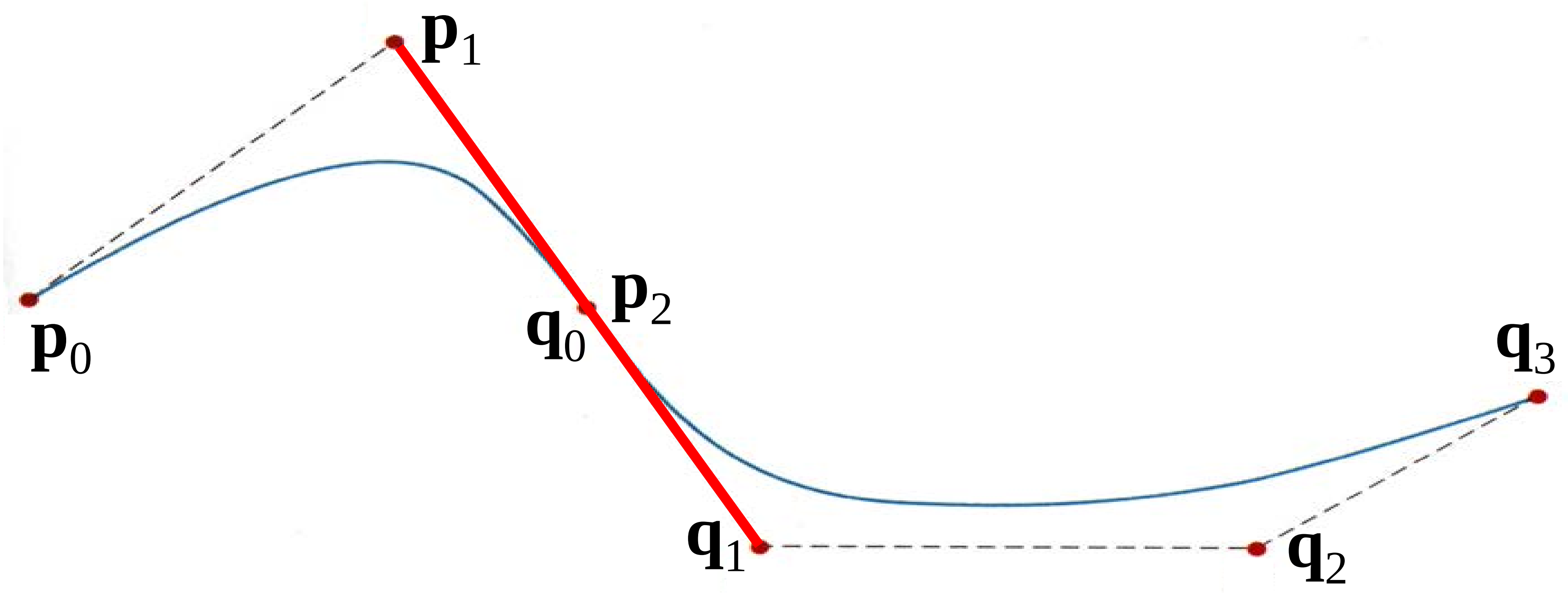
a ***closed Bézier curve*** generated by setting: first = last control point




a Bézier curve can be made to pass closer to a given coordinate position by assigning ***multiple control points*** to that position



# Bézier Curves Design Techniques (2)




*piecewise approximation curve* formed with 2 Bézier sections.
0-order and 1st-order continuity ($C^0$, $C^1$ or $G^0$, $G^1$) are attained by setting $\mathbf{q}_0 = \mathbf{p}_2$ and by making $\mathbf{p}_1$, $\mathbf{p}_2$, and $\mathbf{q}_1$ collinear.

# Cubic Bézier Curve Matrix Notation

$$\mathbf{p}(u) = (1-u)^3 \cdot \mathbf{p_0} + 3u(1-u)^2 \cdot \mathbf{p_1} + 3u^2(1-u) \cdot \mathbf{p_2} + u^3 \cdot \mathbf{p_3}$$

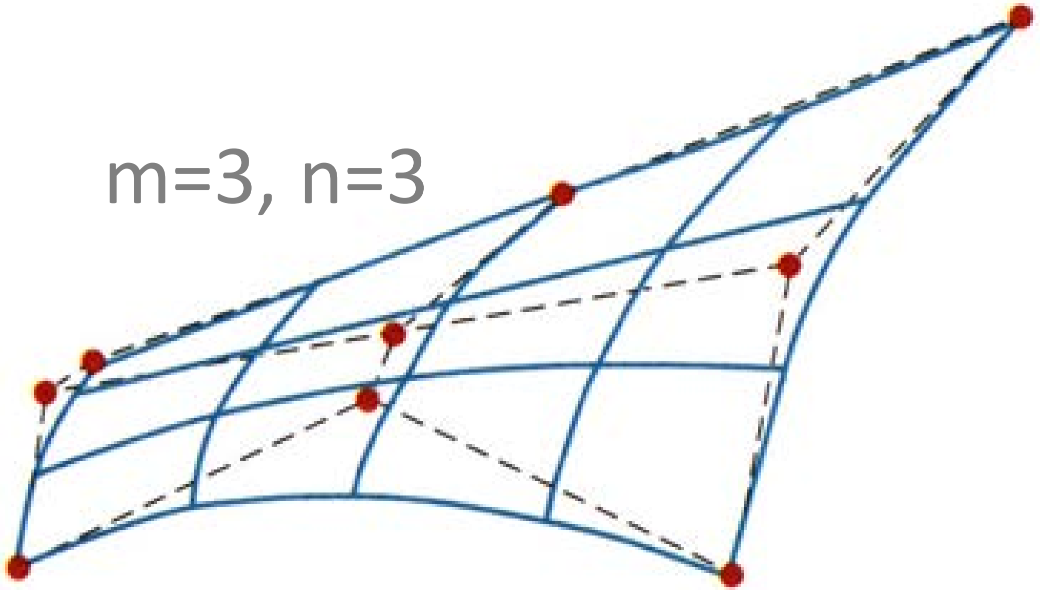
$$\mathbf{p}(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \cdot M_{Bez} \cdot \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix} \quad \text{with} \quad M_{Bez} = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

# Bézier Surfaces Definition

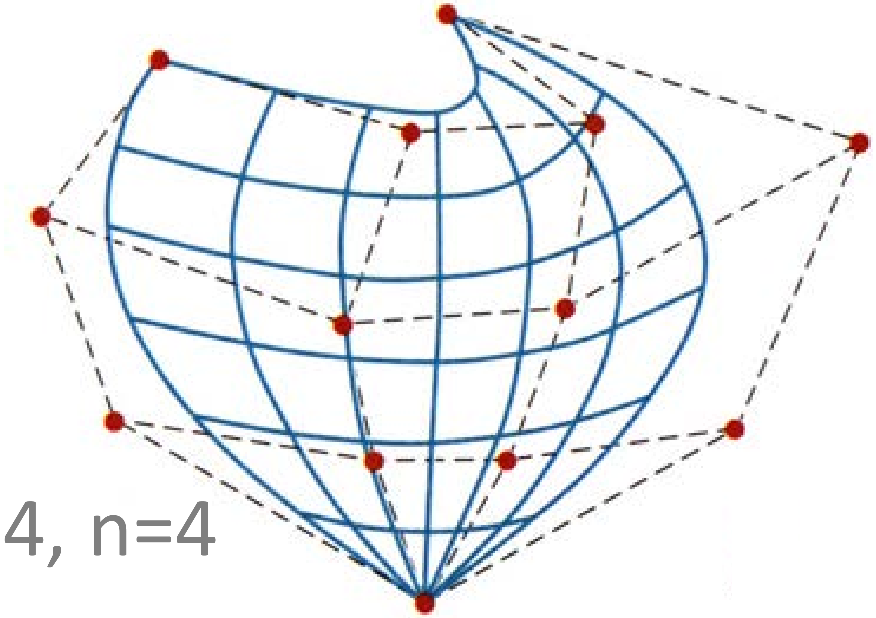
= Cartesian product of two Bézier curve bundles

$$\mathbf{p}(u,v) = \sum_{j=0}^{m} \sum_{k=0}^{n} \mathbf{p}_{\mathbf{j,k}} b_{\mathbf{j,m}}(v)\, b_{\mathbf{k,n}}(u)$$

$\mathbf{p}_{\mathbf{j,k}}$ : grid of (m+1)x(n+1) control points



m=3, n=3

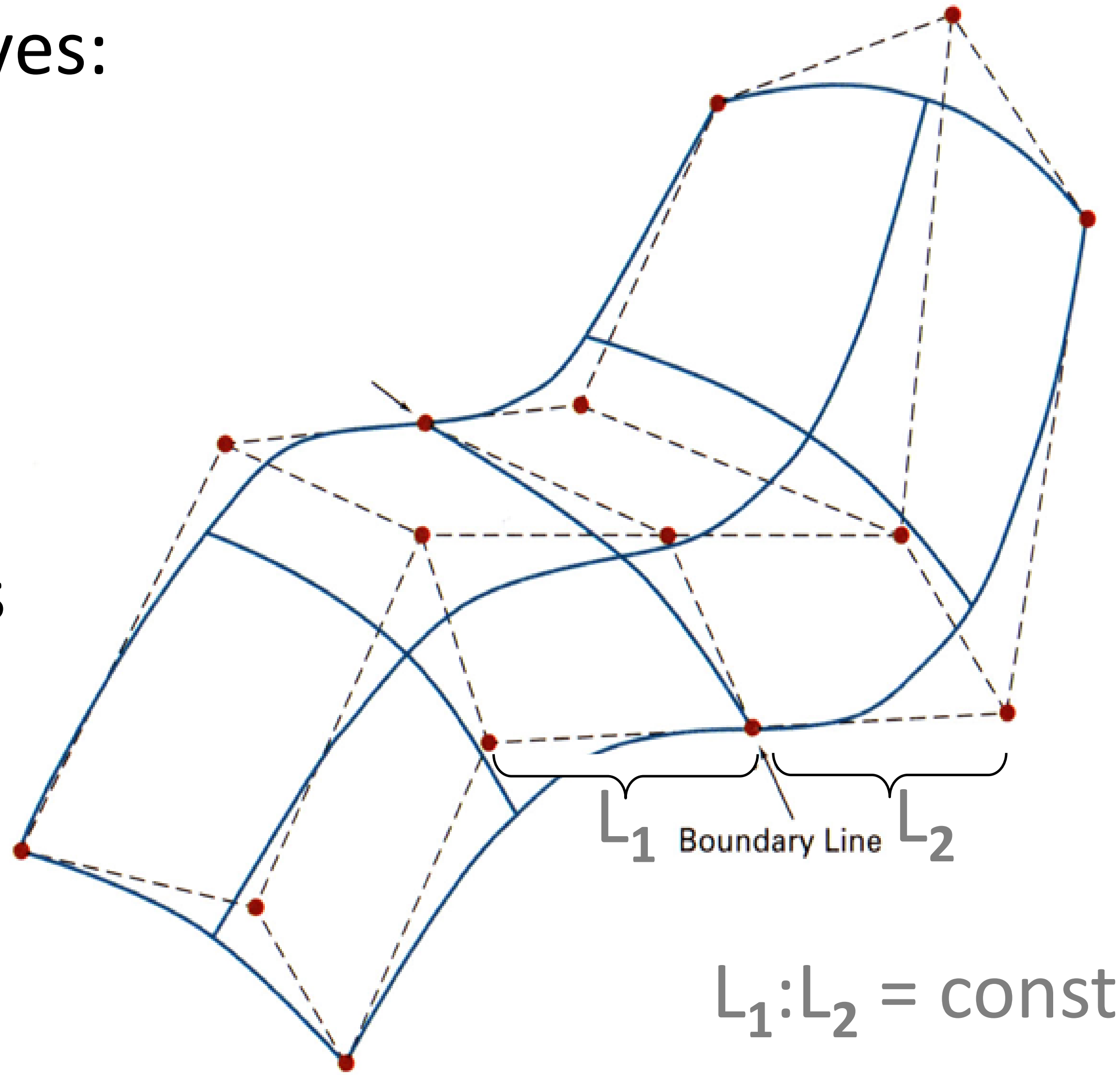


m=4, n=4

# Bézier Surfaces Properties

the same properties as Bézier curves:

- global influence
- interpolates corner points
- tangents at corner points
- convex hull property
- 1st-order continuity connections




$L_1:L_2$ = constant

# B-Spline Curves and Surfaces

= spline approximation for points $\mathbf{p_i}$ , $i=0,\ldots, n$

$$\mathbf{p}(u) = \sum_{k=0}^{n} \mathbf{p}_k B_{k,d}(u) \qquad u_{min} \leq u \leq u_{max} \qquad 2 \leq d \leq n+1$$

B-Spline blending functions from recursive Cox-deBoor formulas

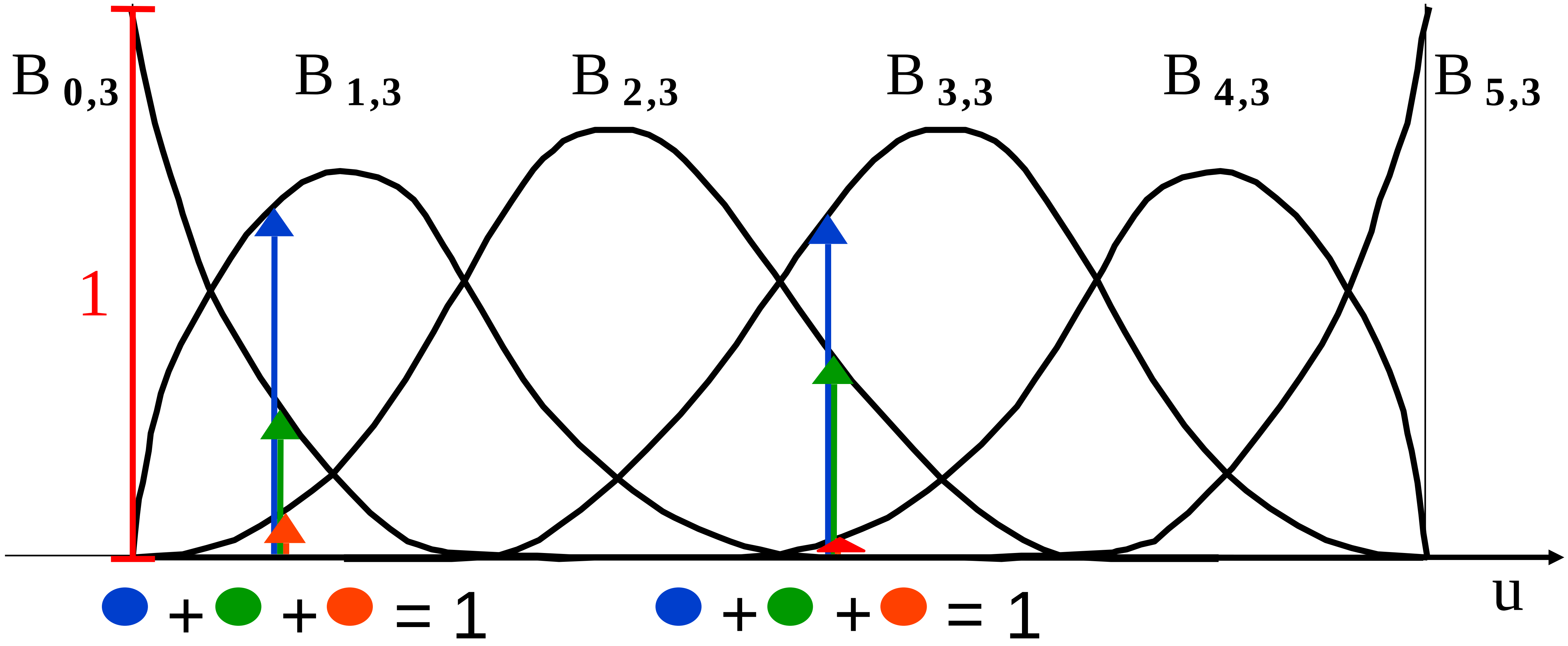
# B-Spline Basis Functions

$$B_{k,1}(u) = \begin{cases} 1 & \text{if} \quad u_k \le u \le u_{k+1} \\ 0 & \text{otherwise} \end{cases}$$

$$B_{k,d}(u) = \frac{(u - u_k) \cdot B_{k,d-1}(u)}{u_{k+d-1} - u_k} + \frac{(u_{k+d} - u) \cdot B_{k+1,d-1}(u)}{u_{k+d} - u_{k+1}} \qquad \text{for } 0 \le u \le n-d+2$$

$$u_k = \left\{ \begin{array}{lll} 0 & \text{for} & k < d \\ k-d+1 & \text{for} & d \le k \le n \\ n-d+1 & \text{for} & k > n \end{array} \right\} \text{global, do not change}$$

# B-Spline Basis Functions for d=3



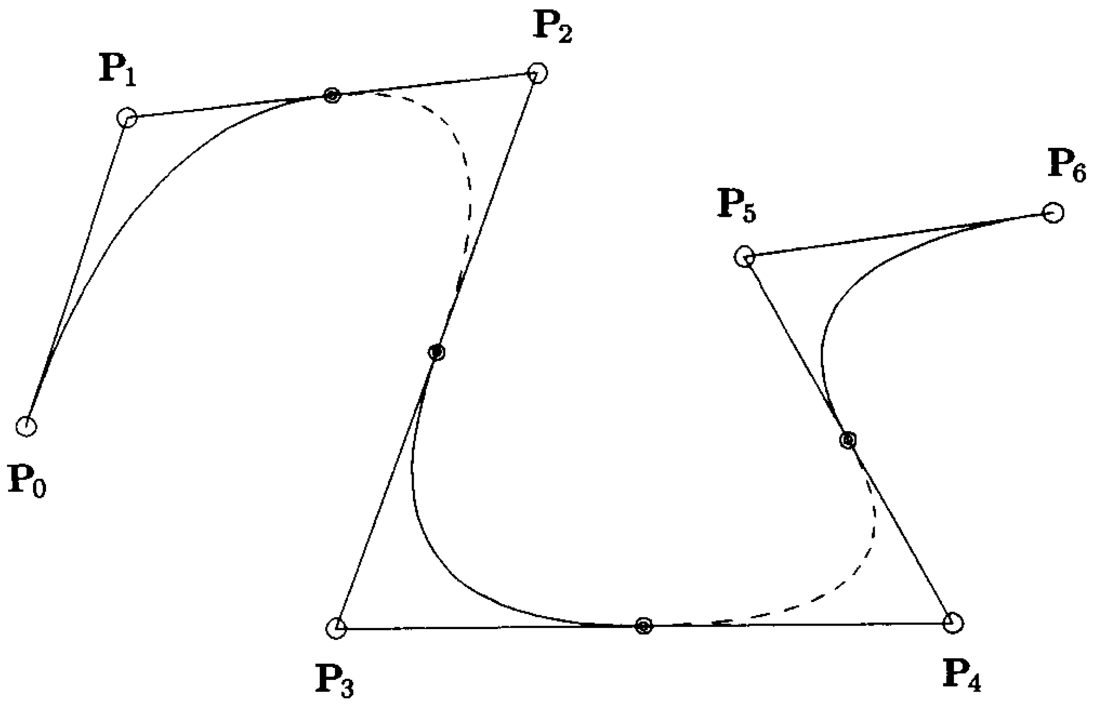
# Important Property of the $B_{k,d}$

for all B-Spline basis functions the following property holds:

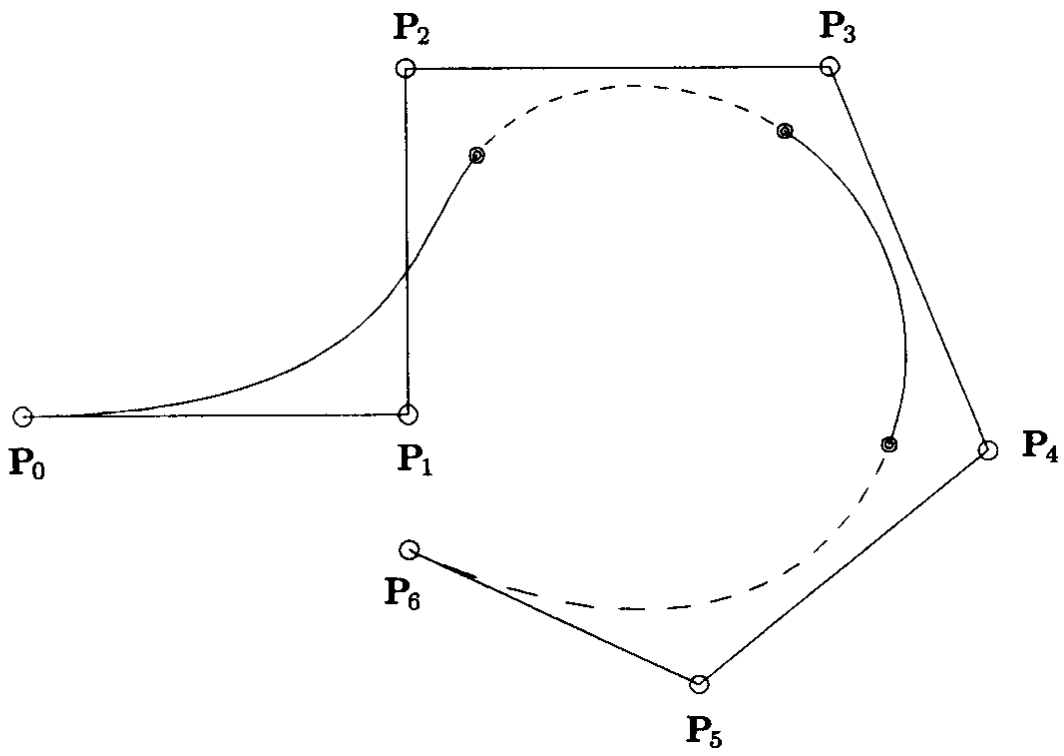
$$\boxed{\sum_{k=0}^{n} B_{k,d}(u) = 1 \qquad \text{for all } u}$$

$\Rightarrow$ every curve point is a weighted mean of the control points

# 2-Dimensional B-Spline Examples




d=3

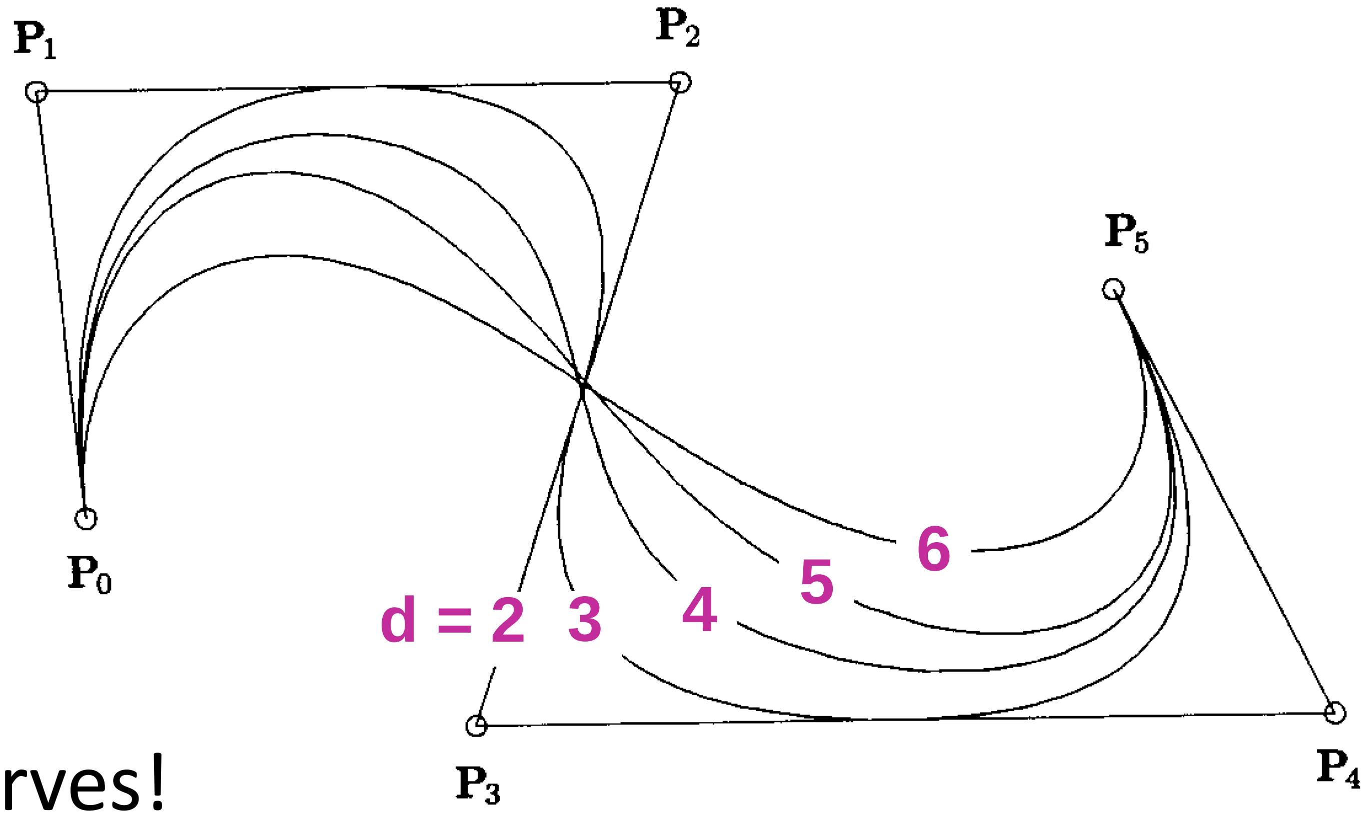



d=4

# Influence of d

d describes, how many control points influence every curve point

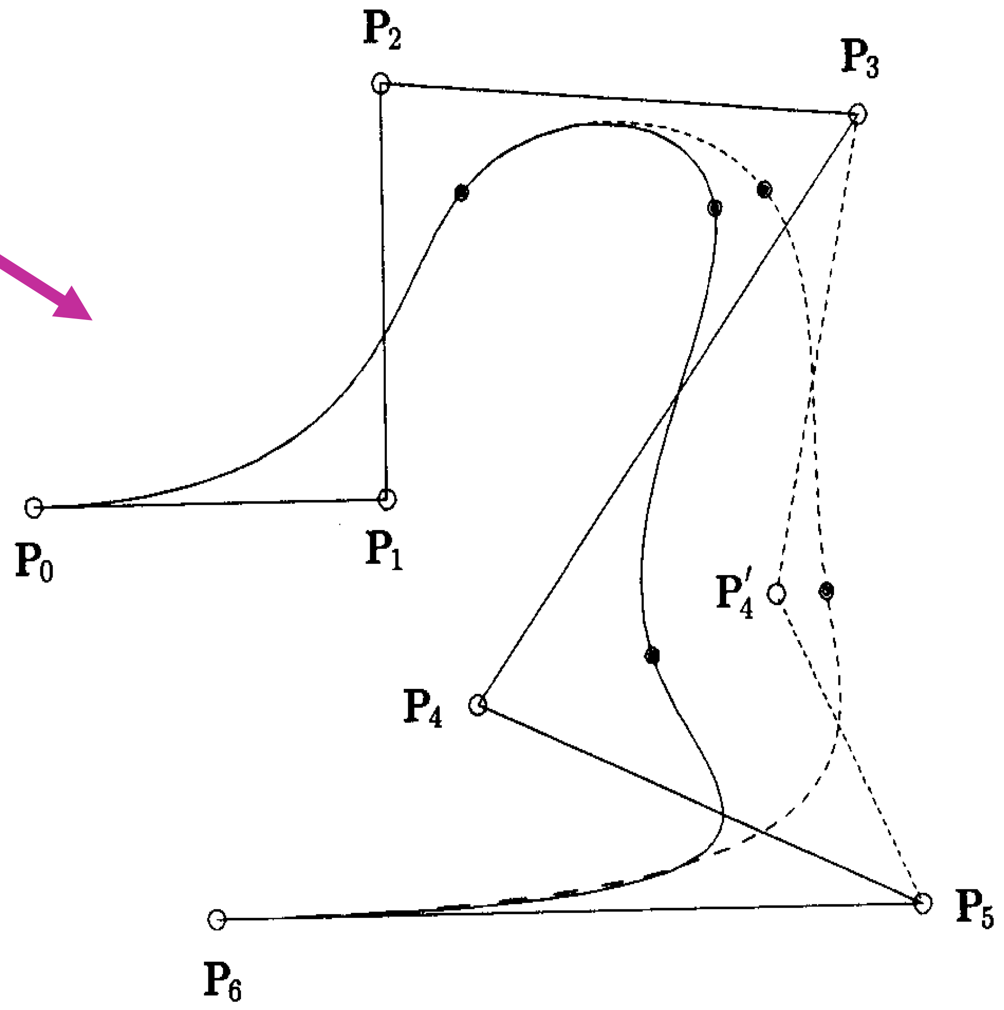
- d = 2 linear
- d = 3 quadratic
- d = 4 cubic
- …




for d=n+1 you get Bézier curves!

# Differences B-Spline ↔ Bézier

control points have local influence



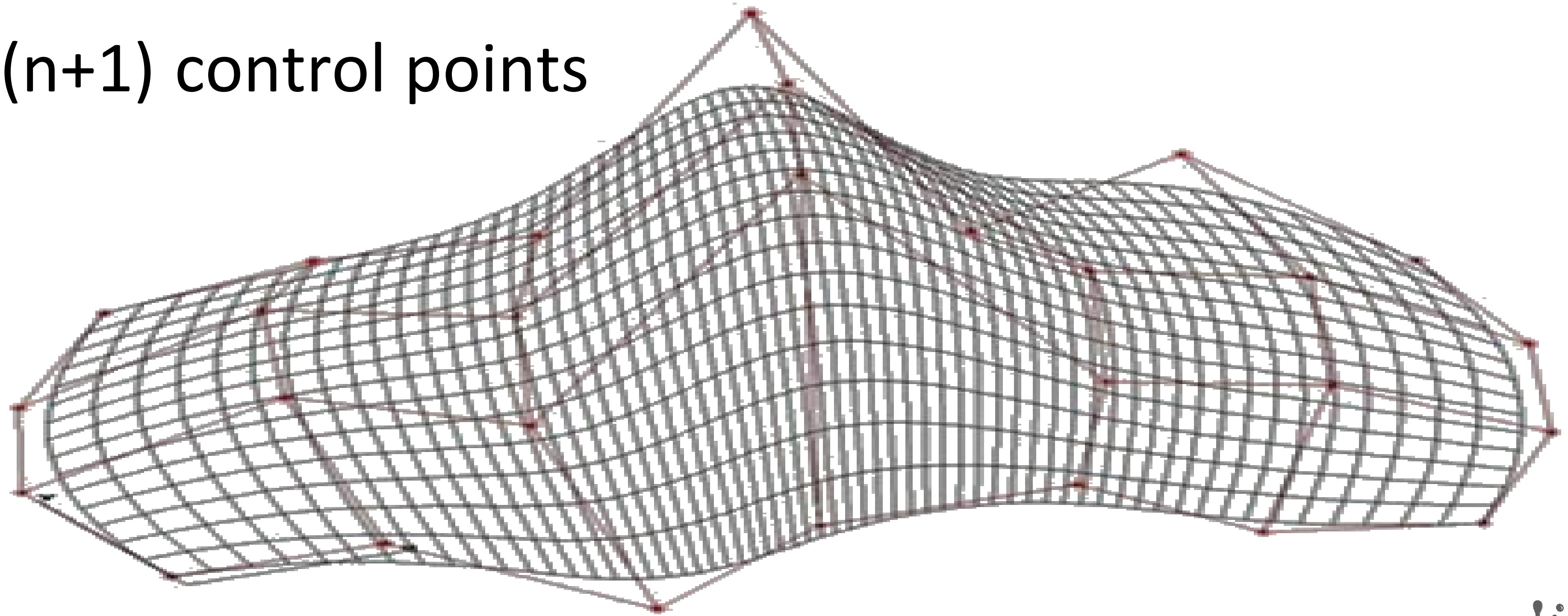

effort is linearly dependent on n, therefore splitting of huge point sets not necessary

= Cartesian product of 2 B-Spline curve bundles

$$\mathbf{p}(u,v) = \sum_{j=0}^{m} \sum_{k=0}^{n} \mathbf{p_{j,k}} \, B_{j,d}(u) \, B_{k,d}(v)$$

$\mathbf{p_{j,k}}$ : grid of (m+1)x(n+1) control points

just like with
Bezier surfaces!



further extension:

**N**on**U**niform **R**ational **B**-**S**plines = “NURBS”

allow to combine freeform surfaces with regular surfaces